Rapport sur l'épreuve écrite d'Informatique 2008

ENS: Cachan Lyon Paris
Membres du jury: S. Demri, C.-P. Jeannerod, P. Letouzey

Juillet 2008

1 Présentation du sujet

Le sujet s'intéressait à la structure cyclique de certains graphes. La première partie portait sur le calcul d'un arbre couvrant et des cycles fondamentaux d'un graphe non orienté. Les questions posées dans cette partie concernaient des algorithmes relativement classiques, excepté l'algorithme de la question 1.3(3).

La seconde partie permettait d'étudier plusieurs algorithmes de calcul d'une base minimale de l'espace des cycles d'un graphe non orienté et pondéré. Dans un premier temps, on donnait un tel algorithme et il s'agissait de l'analyser puis d'en proposer une version plus rapide (questions 2.1 à 2.4). Dans un deuxième temps et pour finir, la question 2.5 (longue et difficile) permettait de retrouver un algorithme récent de Mehlhorn et Michail utilisant le produit de matrices.

Cette épreuve d'une durée de 4 heures avait pour but de tester l'aptitude des candidats à:

- concevoir des algorithmes simples et écrire les pseudo-programmes associés (voir par exemple la question 1.1(3)),
- analyser la complexité des algorithmes (voir par exemple la question 2.4(3)) et utiliser l'approche "diviser pour régner" (voir par exemple la question 2.5(1)),
- raisonner sur des graphes à l'aide d'espaces vectoriels (voir la deuxième partie).

2 Remarques générales

Algorithmes On rappelle que l'objectif pour la production de pseudo-code n'est pas seulement d'être correct, mais aussi et surtout d'être convaincant pour le correcteur. Trop de choses difficilement lisibles, obscures, sans réelles explications ni justifications. Attention également à la longueur: en règle générale, les algorithmes demandés dans ce genre d'épreuve peuvent tenir sur une seule page.

Consignes Un manque flagrant de respect des consignes, allant de la simple étourderie à la tentative de détourner les questions à son avantage. En particulier:

- À la question 1.1(1a), on ne demandait aucune borne, juste la finitude. Pourquoi se lancer alors dans une majoration fine et donc risquée ?
- À la question 1.2(1), on demandait la structure d'adjacence de l'arbre couvrant de de G et non de G lui-même.
- Dans le pseudo-code, l'utilisation de `Adj` au lieu de (S, A) n'était en rien facultative. Si certains candidats souhaitaient vraiment utiliser la liste de toutes les arêtes, il fallait alors dire comment l'obtenir depuis `Adj`.
- À la question 1.3(2b), l'expression "ordre de grandeur" était cruciale. Dire que $9 \neq 10$ sur un cas particulier ou même que $n^2 \neq n^2/2$ n'était en rien satisfaisant.
- À la question 2.3(1a), le pseudo-programme demandé devait en particulier garantir $S_i \neq 0$, une condition nécessaire pour que S_i soit un support.
- Dans les calculs de complexité, attention à fournir un résultat en fonction des variables demandées; également, en répondant aux questions du type "quel est le coût de cet algorithme", chercher à donner une majoration asymptotique la plus fine possible (on a souvent vu des $O(2^n)$ pour des algorithmes de coût quadratique ou linéaire en n).
- Pour certaines questions comme 1.3(1) et 1.3(2b), ne pas hésiter à faire des dessins en complément d'une justification concise.

3 Informations quantitatives

La moyenne globale s'établit à 10 sur 20 avec un écart-type de 4. La note minimale est 1,6 et la note maximale est 20. Traiter uniquement la première partie permettait de recevoir la note de 13. Traiter les questions 2.1, 2.2 et 2.3 de la deuxième partie rapportait plus de 7 points.

On donne ci-dessous, pour chaque question, le pourcentage de candidats ayant eu une note strictement positive à cette question :

Partie 1 :

1.1(1a) :	65%
1.1(1b) :	83%
1.1(2) :	76%
1.1(3) :	65%
1.2(1) :	60%
1.2(2) :	46%
1.3(1) :	63%
1.3(2a) :	65%
1.3(2b) :	53%
1.3(3a) :	24%
1.3(3b) :	11%

Partie 2 :

2.1(1a) :	65%
2.1(1b) :	53%
2.1(2) :	63%
2.1(3) :	42%
2.2(1) :	53%
2.2(2) :	52%
2.3(1a) :	9%
2.3(1b) :	4%
2.3(2a) :	5%
2.3(2b) :	3%
2.4(1) :	43%
2.4(2a) :	18%
2.4(2b) :	3%
2.4(2c) :	3%
2.4(3a) :	6%
2.4(3b) :	3%
2.5(1a) :	3%
2.5(1b) :	5%
2.5(2a) :	1%
2.5(2b) :	1%
2.5(3a) :	0%
2.5(3b) :	0%